How Does a Deep Learning Model Architecture Impact Its Privacy?
A Comprehensive Study of Privacy Attacks on CNNs and Transformers

Guangsheng Zhang[1] Bo Liu[1] Huan Tian[1] Tianqing Zhu[1]
Ming Ding[2] Wanlei Zhou[3]

[1] *University of Technology Sydney* [2] *Data 61, Australia* [3] *City University of Macau*

Abstract

As a booming research area in the past decade, deep learning technologies have been driven by big data collected and processed on an unprecedented scale. However, privacy concerns arise due to the potential leakage of sensitive information from the training data. Recent research has revealed that deep learning models are vulnerable to various privacy attacks, including membership inference attacks, attribute inference attacks, and gradient inversion attacks. Notably, the efficacy of these attacks varies from model to model. In this paper, we answer a fundamental question: *Does model architecture affect model privacy?* By investigating representative model architectures from convolutional neural networks (CNNs) to Transformers, we demonstrate that Transformers generally exhibit higher vulnerability to privacy attacks than CNNs. Additionally, we identify the micro design of activation layers, stem layers, and LN layers, as major factors contributing to the resilience of CNNs against privacy attacks, while the presence of attention modules is another main factor that exacerbates the privacy vulnerability of Transformers. Our discovery reveals valuable insights for deep learning models to defend against privacy attacks and inspires the research community to develop privacy-friendly model architectures.

1 Introduction

Deep learning has been gaining massive attention over the past several years. Training deep learning models requires collecting and processing user data, which raises significant privacy concerns. The data gathered during the training phase often contains sensitive information that malicious parties can access or retrieve. Various privacy attacks targeting deep learning models have demonstrated this vulnerability extensively. One prominent type of attack is membership inference, which focuses on determining whether a specific data sample belongs to the training data [60,62]. Another attack is attribute inference, which aims to uncover implicit attributes learned by the model beyond the intended target attribute [52,65]. Additionally, gradient inversion attacks pose a significant threat by attempting to reconstruct the information of the training data from the gradients of the model [20,22]. These attacks empower adversaries to exploit deep learning models for extracting sensitive data.

Prior research has established that overfitting is one of the primary causes of privacy leakage in deep learning models [10,27,43,62]. In general, overfitting occurs when models excessively learn specific details from the training data, which can lead to inadvertent privacy breaches. Surprisingly, we discover that even when models exhibit comparable levels of overfitting, the effectiveness of attacks varies across different models. This observation raises intriguing questions as to why certain deep learning models are more susceptible to privacy attacks than others, a puzzle that researchers have not fully comprehended. Consequently, we conjecture that other factors beyond overfitting might also contribute to the increased vulnerability of some deep learning models to privacy attacks. Though existing literature has explored model robustness and explainability [4,57], the privacy leakage of the model architectures remains underexplored. Therefore, we are motivated to address this critical gap by answering the following question: *How does a model's architecture affect its privacy preservation capability?*

In this paper, we approach this question by comprehensively analyzing different deep learning models under various state-of-the-art privacy attacks. Our investigation focuses on two widely adopted deep learning model architectures: convolutional neural networks (CNNs) and Transformers. CNN-based models have been dominant in computer vision, thanks to its sliding-window strategy, which extracts local information from images effectively. Transformers, initially introduced in natural language processing (NLP), have gained popularity in computer vision by capturing large receptive fields through attention mechanisms, resulting in comparable accuracy performance against CNNs. The tremendous achievements and wide usage of these two model architectures provide an excellent opportunity for us to make a comparative analysis regarding model privacy risks. Through our investigation, we make an intriguing discovery: *Transformers,* *in general, exhibit higher vulnerability to mainstream privacy attacks than CNNs.*

While Transformers and CNNs have different designs in many aspects, we investigate whether some key modules in the model architecture have a major impact on privacy risks. To this end, we evaluate the privacy leakage of several major modules in a Transformer architecture by sending only selected gradients to the gradient inversion attacks and discover that attention modules cause significant privacy leakage. Moreover, we start with a popular CNN-based model, ResNet-50 [26], and gradually morph the model to incorporate the key designs of Transformers. This leads us to the structure of ConvNeXt [46]. We evaluate the privacy leakage through this process and identify several key components that have a significant impact on privacy risks: (1) the design of the activation layers; (2) the design of stem layers; (3) the design of LN layers. We further conduct ablation studies to verify our discoveries and propose solutions to mitigate the privacy risks.

In summary, our contributions in this paper are summarized as follows:

- For the first time, we investigate the impact of model architectures and micro designs on privacy risks.
- We evaluate the privacy vulnerabilities of two widely adopted model architectures, i.e., CNNs and Transformers, using three prominent privacy attack methods: (1) membership inference attacks, (2) attribute inference attacks, and (3) gradient inversion attacks. Our analysis reveals that Transformers exhibit higher vulnerabilities to these privacy attacks than CNNs.
- We identify three key factors: (1) the design of activation layers, (2) the design of stem layers, and (3) the design of LN layers, that significantly contribute to the enhanced resilience of CNNs in comparison to Transformers. We also discover that the presence of attention modules in Transformers could make them susceptible to privacy attacks.
- We propose solutions to mitigate the vulnerabilities of model architectures: modifying model components and adding perturbations as defense mechanisms.

2 Related Work

2.1 CNNs and Vision Transformers

Convolutional Neural Networks (CNNs) are a type of neural network that employs convolutional layers to extract features from input data. In contrast to fully connected networks, CNNs use convolutional kernels to connect small samples to neurons for feature extraction, reducing the number of model parameters and enabling the recognition of local features. Various techniques are employed to construct a CNN model, including padding, pooling, dilated convolution, group convolution, and more.

The concept of convolutional neural networks (CNNs) dates back to the 1980s [38]. However, the invention of AlexNet [37] makes CNNs the most prominent models in computer vision. Subsequent research improved the accuracy and efficiency of models [63, 68]. ResNet [26] addressed the challenge of training deep networks using skip connections. Other notable networks consist of Inception [69], MobileNet [30], ResNeXt [79], EfficientNet [70], RegNet [56], ConvNeXt [46].

Vision Transformers, originating from natural language processing, divide the input image into multiple patches, forming a one-dimensional sequence of token embeddings. Their exceptional performance can be attributed to the multi-head self-attention modules [74]. The attention mechanism has significantly contributed to the advancement of natural language processing [5, 14, 81], subsequently leading to the introduction of Transformers in the field of computer vision as Vision Transformers (ViT) [16]. Research has shown that ViTs can surpass CNNs in various downstream tasks [16, 67]. Later Transfomer models have focused on numerous improvements of ViTs, such as Tokens-to-Token ViT [85], Swin Transformers [45], DeiT [71], MViT [40], DaViT [15].

Numerous studies have compared CNNs and Transformers from the perspectives of robustness [4, 55, 75] and explainability [57]. However, our research diverges from previous works by concentrating on the privacy leakage inherent in both CNNs and Transformers.

2.2 Privacy Attacks on Deep Learning Models

A primary concern in deep learning privacy is that the model may reveal sensitive information from the training dataset. An adversary can exploit various approaches to compromise privacy, including predicting whether a particular sample is in the model's training dataset via membership inference attacks, or disclosing the implicit attributes of data samples via attribute inference attacks, or even recovering private data samples utilized in training a neural network through gradient inversion attacks.

Membership inference attacks were initially introduced in [62], where an attack model was employed to distinguish member samples from non-member samples in the training data. To execute these attacks, shadow models would mimic the behavior of victim models [60, 62]. Prediction results from victim models were gathered for attack model training. Usually, the confidence scores or losses were utilized [62], but more recent work (label-only attacks) applied prediction labels to launch attacks successfully [11, 41]. The attacks could also be executed by designing a metric with a threshold by querying the shadow model [66]. Some researchers expanded the attacks into new domains, including generative models [9, 25], semantic segmentation [28, 87], federated learning [54, 73], and transfer learning [64, 94]. Other researchers relaxed the attack assumptions and improved the attacks, including discussion on white-box/black-box access for the attacks [59], providing more metrics (i.e. ROC curves and the true positive rate at a low false positive rate) to measure the attack performance more accurately [7,34,48,77,82]. We select [7,60,62] as our baseline methods.

Attribute inference attacks, another significant category of privacy attack methods, attempt to reveal a specific sensitive attribute of a data sample by analyzing the posteriors of the victim model trained by the victim dataset. Some early research launched the attacks by generating input samples with different sensitive attributes and observed the victim model output [21,83]. However, these methods could only work in structured data. Later research improved the attacks with victim model representations [52,65]. They also claimed that the overlearning feature of deep learning models caused the execution of the attacks [65]. Attributes could also be inferred through a relaxed notion [91], model explanations [18], label-only settings [51], or imputation analysis [33]. As we aim to infer attributes from visual data, we select [52,65] as baseline methods.

Gradient inversion attacks primarily aim to reconstruct training samples at the local clients in federated learning. Using the publicly shared gradients in the server, adversaries can execute the attacks by reconstructing the training samples using gradient matching. DLG [93] and its variant, iDLG [92], were the early attacks to employ an optimization-based technique to reconstruct the training samples. Later research like Inverting Gradients [22] and GradInversion [84] improved the attack performance by incorporating regularizations into the optimization process. APRIL [49] and GradViT [24] further developed the attack methods to extract sensitive information from Transformers. The use of Generative Adversarial Networks (GANs) in some gradient inversion attack methods [42] can have a significant impact on reconstructed results, making it difficult to isolate the influence of other factors on privacy leakage. Therefore, we use conventional gradient inversion attack methods [22] that do not involve the use of GANs.

There have been several evaluations and reviews of these privacy attacks against deep learning models [27,31,43,44,66,88,90]. However, we aim to evaluate the model architectures leveraging these privacy attacks. To sum up, we utilize conventional privacy attacks [7,22,52,60,62,65] as the baseline attacks in our analysis, for these attack methods have inspired many follow-up research works, and they are suitable for evaluation on various models and datasets.

3 Methodology of Evaluating the Impact of the Model Architecture on Privacy

In this section, we present our approach to assessing the impact of model architectures on privacy leakage. In order to organize our study in a thorough and logical manner, We aim to answer the following research questions sequentially:

- **RQ1**: How to analyze the privacy leakage in model architectures?
- **RQ2**: What model architectures of CNNs and Transformers should we choose to evaluate these attacks?
- **RQ3**: What performance aspects should we focus on when evaluating the privacy attacks on model architectures?
- **RQ4**: How should we investigate what designs in model architectures contribute to privacy leakage?

In this work, we focus on classifier or feature representation models such as CNNs and Transformers, which are subject to the investigated privacy attacks. A new line of generative AI models, such as generative adversarial networks (GANs) and diffusion models, are vulnerable to different privacy attacks and thus out of the scope of this paper. We believe our evaluation methodology could shed light on the effect of model privacy from the perspective of model architectures.

3.1 Privacy Threat Models

To answer the first research question (**RQ1**), we choose three prominent privacy attack methods: membership inference attacks, attribute inference attacks, and gradient inversion attacks.

3.1.1 Membership Inference Attacks

Network-Based Attacks. Initiating a network-based membership inference attack [60,62] requires three models: the victim model $\mathcal{V}$ (the target), the shadow model $\mathcal{S}$ (the model to mimic the behavior of the victim model), and the attack model $\mathcal{A}$ (the classifier to give results whether the sample belongs to the member or non-member data). The following paragraphs provide explanations of how the attacks work.

The first step is the attack preparation. Since the adversary has only black-box access to the victim model $\mathcal{V}$, they can only query the model and record prediction results. To launch a membership inference attack, the adversary needs to create a shadow model $\mathcal{S}$, which behaves similarly to the victim model $\mathcal{V}$. This involves collecting a shadow dataset $\mathcal{D}_S$, usually from the same data distribution as the victim dataset $\mathcal{D}_V$. The shadow dataset $\mathcal{D}_S$ is then divided into two subsets: $\mathcal{D}_S^{train}$ for training and $\mathcal{D}_S^{test}$ for testing.

Once the preparation is complete, the adversary trains the attack model. The shadow model $\mathcal{S}$ and shadow dataset $\mathcal{D}_S$ are used to train the attack model $\mathcal{A}$. Each prediction result of a data sample from the shadow dataset $\mathcal{D}_S$ is a vector of confidence scores for each class, which is concatenated with a binary label indicating whether the prediction is correct or not. The resulting vector, denoted as $\mathcal{P}_S^i$, is collected for all n samples, forming the input set $\mathcal{P}_S = \{\mathcal{P}_S^i, i = 1, ..., n\}$ for the attack model $\mathcal{A}$. Since $\mathcal{A}$ is a binary classifier, a three-layer MLP (multi-layer perceptron) model is employed to train it.

At last, the adversary launches the attack model inference. The adversary queries the victim model $\mathcal{V}$ with the victim dataset $\mathcal{D}_V$ and records the prediction results, which are used as the input for the attack model $\mathcal{A}$. The attack model then predicts whether a data sample is a member or non-member data sample.

Likelihood-Based Attacks. The Likelihood Ratio Attack (LiRA) [7] is a state-of-the-art attack method that employs both model posteriors and their likelihoods based on shadow models. In contrast to attacks relying on a single shadow model, LiRA requires the adversary to train multiple shadow models $\mathcal{S} = \{S_1, ..., S_n\}$. This ensures that a target sample (from the victim dataset $\mathcal{D}_V$) is included in half of the models $\mathcal{S}$ and excluded from the other half. The adversary then queries the shadow models with the target sample and calculates the logits for each model. Using these logits, the adversary calculates the probability density function to determine the likelihood ratio of the target sample, which corresponds to its membership status.

There are other kinds of membership inference attacks, including metric-based attacks and label-only attacks [11, 41, 66]. Instead of using a neural network to be the attack model, metric-based attacks [66] launch the attacks using a certain metric and threshold to separate member data from non-member data. Label-only attacks [11, 41] relax the assumptions of the threat model leveraging only prediction labels as the input of the attack model. Our study focuses on two types of membership inference attacks: network-based and likelihood-based attacks. We chose these two types of attacks because the network-based attack is commonly used as a baseline in many research papers, making it a conventional attack to consider. Additionally, the likelihood-based attack is a more recent state-of-the-art attack that has demonstrated high effectiveness, making it an important attack to evaluate as well. By considering these two types of attacks, we can effectively represent the performance of membership inference attacks against various victim models and gain insights into potential privacy risks associated with different machine learning models.

3.1.2 Attribute Inference Attacks

The goal of attribute inference attacks [52, 65] is to extract sensitive attributes from a victim model, which may inadvertently reveal information about the training data. For instance, suppose the victim model is trained to classify whether a person has a beard or not. In that case, an adversary may infer the person's race based on the model's learned representation.

At the attack preparation stage, the victim model $\mathcal{V}$ is trained by the victim dataset $\mathcal{D}_V$ with two subsets $\mathcal{D}_V^{train}$ and $\mathcal{D}_V^{test}$ for the training and testing.

The second step is also the attack model training. To train the attack model $\mathcal{A}$, the adversary uses an auxiliary dataset $\mathcal{D}_A^{train}$, which includes pairs of the representation h and the attribute a, i.e., $(h, a) \in \mathcal{D}_A$.

At last, the adversary launches the attack. The adversary takes a data sample's representation h as the input and uses the attack model $\mathcal{A}$ to infer the attribute result.

3.1.3 Gradient Inversion Attacks

Launching the gradient inversion attack [22, 92, 93] involves solving an optimization problem, which aims to minimize the difference between the calculated model gradients and the original model gradients. The optimization process continues for a certain number of iterations, after which the input data sample can be reconstructed.

The adversary operates within a federated learning scenario. In the attack preparation stage, the adversary operates from the central server, aggregating model gradients to create a centralized model. Since the adversary has access to the communication channels used during the federated learning process, they can retrieve the model gradients and prepare to extract sensitive information from the training samples. This allows the adversary to launch attacks against the federated learning system.

In the step of gradient reconstruction, the aggregated model gradients are denoted as $\nabla_\theta \mathcal{L}_\theta(x, y)$, where θ is the model parameters, x and y are the original input image and its ground truth in a local client, and $\mathcal{L}$ represents the cost function for the model. To initiate the reconstruction process, the adversary generates a dummy image x^*. The adversary tries to minimize this cost function: $\arg\min_x ||\nabla_\theta \mathcal{L}_\theta(x, y) - \nabla_\theta \mathcal{L}_\theta(x^*, y)||^2$. The dummy image x^* is reconstructed to resemble x closely.

3.2 CNNs vs Transformers

To answer the second research question (**RQ2**), we investigate the privacy of two mainstream architectures: CNNs and Transformers. We carefully select several popular CNNs and Transformers for the attacks to analyze the privacy leakage.

For CNNs, we choose ResNets [26] as baseline models, which are known for incorporating residual blocks and have become widely used in various computer vision tasks. We specifically select ResNet-50 (23.52 million parameters) and ResNet-101 (42.51 million parameters) to represent CNN architectures in our analysis. Regarding Transformers, we focus on Swin Transformers [45], which have gained attention for their innovative design incorporating attention modules and shifted window mechanisms. We analyze Swin-T (27.51 million parameters) and Swin-S (48.80 million parameters) as representatives of Transformer architectures.

To ensure fair comparisons, we organize the evaluation of the four models based on their parameter sizes, grouping models with similar parameter sizes together. This approach allows us to compare models that exhibit comparable task

Table 1: Training recipes for privacy attacks.

Config	Param	Config	Param
for all models of membership inference			
optimizer	AdamW	training epochs	300
learning rate	0.001	batch size	256
weight decay	0.05	mixup	0.8
optimizer momentum	$\beta_1, \beta_2 = 0.9, 0.999$	cutmix	1.0
learning rate schedule	cosine annealing	label smoothing	None
for all models of attribute inference			
optimizer	SGD	batch size	256
learning rate	0.01	training epochs	100
weight decay	0.0005	random augmentation	None
optimizer momentum	0.9		
for gradient inversion			
cost function	similarity	total iteration	3000
optimizer	Adam	total variance	0.0001
learning rate	0.1		

performances while considering their architectural differences. Specifically, we compare ResNet-50 with Swin-T, as they have similar parameter sizes, and we compare ResNet-101 with Swin-S for the same reason.

Apart from model architectures, training Transformers requires a modernized training procedure compared to training traditional CNNs [45, 74]. To ensure fair comparisons, we employ the same training recipe for both CNNs and Transformers in each comparison.

4 Evaluation on Privacy Attacks with CNNs and Transformers

This section addresses the third research question (**RQ3**) and comprehensively analyzes the experimental settings and results. We aim to compare the attack performance under various metrics to gain a deeper understanding of the findings. We also focus on the performance differences and overfitting differences between CNNs and Transformers.

4.1 Settings

Datasets. Our experiments evaluate the privacy leakage under these four datasets.

- **CIFAR10 [36]** consists of 60,000 color images with dimensions of 32×32 pixels. It is organized into ten classes, with each class containing 6,000 images. The dataset covers a wide range of general object categories.
- **CIFAR100 [36]** is similar to CIFAR10 but with 100 classes. Each class has 600 images and represents a specific general object category.
- **ImageNet1K [13]** is a widely used dataset in computer vision containing over 1 million labeled images, covering 1,000 different classes. The dataset encompasses a diverse range of objects and scenes.
- **CelebA [47]** contains over 200,000 face images, each with 40 binary attributes.

In membership inference attacks, we use two datasets, CIFAR10 and CIFAR100, for their popularity in prior attack research [7, 44, 60]. For network-based attacks, each dataset is evenly split into four subsets for training and testing both the victim and shadow models to ensure a fair evaluation [60]. For likelihood-based attacks, we follow the training settings in [7]. This involves dividing the training data into multiple subsets, ensuring that a specific training sample is present in only half of these subsets.

For attribute inference attacks, we utilize the CelebA dataset, which provides rich attribute labeling. This allows us to identify hidden attributes accurately. In our experiments, we focus on inferring the race attribute while using the gender attribute as the classification goal for the victim model. We randomly select 20,000 images from CelebA and evenly split them into four subsets for training and testing both the victim model and the attribute inference attack model.

In gradient inversion attacks, we employ the CIFAR10 and ImageNet1K datasets for low/high-resolution reconstruction. These attacks are conducted in a federated learning scenario to reconstruct the training batch. We randomly select a subset of images from these datasets to evaluate the attacks, ensuring a representative assessment of the attack performance.

In a nutshell, all these datasets are benchmark datasets for evaluating privacy attacks [7, 22, 27, 44, 84]. These datasets cover a wide range of objects with different numbers of classes, making them sufficient for our evaluation.

Victim Models. To ensure fair comparisons, we evaluate the performances of CNNs and Transformers with similar parameter sizes. For all attacks, we select two groups of models based on their parameter sizes. These groups include ResNet-50 and ResNet-101 as CNN-based models and Swin-T and Swin-S as Transformer-based models. We train these models to reach more than 0.99 training accuracy and output testing accuracy (i.e., task accuracy) results in the experiments.

Attack Models. For membership inference attacks, we employ a three-layer MLP model to infer membership information. Attribute inference attacks utilize a two-layer MLP model to uncover learned representations and potentially infer sensitive information. In gradient inversion attacks, we optimize input and generate gradients to reconstruct the original data, revealing private information.

Evaluation Metrics. We evaluate the performance of different attacks using specific metrics. For membership inference attacks, we consider attack accuracy (↑), ROC curve (↑), AUC (↑), and TPR at low FPR (↑). For attribute inference attacks, we assess effectiveness using attack accuracy (↑) and macro-F1 score (↑). Regarding gradient inversion attacks, we use multiple metrics to evaluate the quality of reconstruction results. These metrics include mean square error (MSE ↓), peak signal-to-noise ratio (PSNR ↑), learned perceptual im-

Table 2: Results for network-based membership inference attacks.

	CIFAR10		CIFAR100	
	Task acc ↑	Attack acc ↑	Task acc ↑	Attack acc ↑
ResNet-50	0.8220 ± 0.0023	0.6385 ± 0.0078	0.5288 ± 0.0083	0.8735 ± 0.0029
Swin-T	0.8335 ± 0.0042	0.6904 ± 0.0052	0.5632 ± 0.0056	0.9340 ± 0.0030
ResNet-101	0.8301 ± 0.0037	0.6317 ± 0.0063	0.5313 ± 0.0074	0.8607 ± 0.0034
Swin-S	0.8258 ± 0.0039	0.6405 ± 0.0075	0.5665 ± 0.0059	0.9357 ± 0.0039

age patch similarity (LPIPS ↓) [89], and structural similarity (SSIM ↑) [76]. Note that "↑" means the higher metric corresponds to the higher attack performance, while ↓ means the lower metric leads to the higher attack performance.

Training Settings for Privacy Attacks. Table 1 illustrates the training configurations for membership inference attacks, attribute inference attacks and gradient inversion attacks.

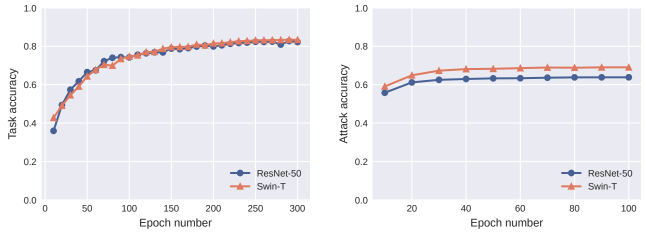
4.2 Evaluation on Membership Inference Attacks

The results presented in Table 2 offer the performance of network-based membership inference attacks on CIFAR10 and CIFAR100 datasets. The task accuracy scores of the victim models on CIFAR10, which are approximately 0.82 for both CNNs and Transformers, indicate that these models exhibit competitive task performance with similar overfitting levels. Notably, the attack accuracy on CIFAR10 reveals that Transformers exhibit more privacy leakage within each group than CNN models. Similar findings are observed on CIFAR100, suggesting that Transformers consistently exhibit higher vulnerability to membership inference attacks compared to CNN models.

It is worth mentioning that the dataset used in this study is divided into four equally sized subsets for training and testing the victim and attack models. Consequently, the task accuracy of the victim models on CIFAR10 and CIFAR100 might be lower than expected in a standard CIFAR10 or CIFAR100 classification task. This does not affect the performance of the attacks, and this phenomenon has been acknowledged in prior research [7, 27, 35, 61]. This also applies to later experimental results.

Figure 1 illustrates the task accuracy and attack accuracy of ResNet-50 and Swin-T across multiple epochs on CIFAR10. The task accuracy of ResNet-50 starts relatively low but exhibits a rapid increase over time. Eventually, both ResNet-50 and Swin-T can achieve task accuracy scores of approximately 0.82. Regarding attack accuracy, the plot demonstrates that the attack on Swin-T consistently outperforms the attack on ResNet-50.

Table 3 displays the outcomes of likelihood-based membership inference attacks on CIFAR10 and CIFAR100. The table includes task accuracy and attack performance, the same as the evaluation for network-based attacks. To assess attack performance more comprehensively, we adopt the methodology proposed in [7] and incorporate additional metrics such as AUC, TPR@0.1%FPR, besides attack accuracy. The victim models achieve task accuracy scores of approximately 0.86 on CIFAR10 and 0.56 on CIFAR100. Consistently, the attack performance metrics highlight that Transformers are more vulnerable to membership inference attacks compared to CNNs in terms of any attack metric. We also present the attack performance through ROC curves in Figure 2. These curves demonstrate that Transformers yield higher ROC curves, signifying their better attack performance.



(a) Performance of victim models on CIFAR10

(b) Performance of privacy attack on CIFAR10

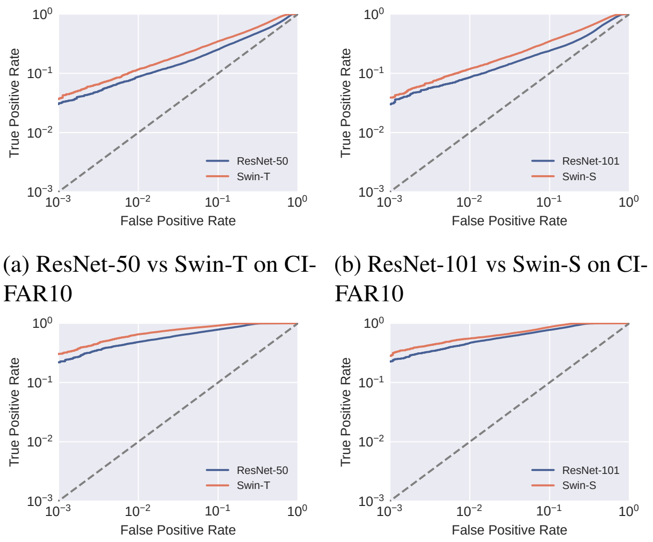
Figure 1: The performance of membership inference attacks against ResNet-50 and Swin-T on CIFAR10 under different numbers of epochs.

Table 3: Results of likelihood-based membership inference attacks.

		Task acc ↑	AUC ↑	TPR@0.1%FPR ↑	Attack acc ↑
CIFAR10	ResNet-50	0.8716 ± 0.0035	0.6446 ± 0.0276	3.15% ± 0.25%	0.6009 ± 0.0163
	Swin-T	0.8630 ± 0.0017	0.7384 ± 0.0029	3.68% ± 0.33%	0.6553 ±0.0027
	ResNet-101	0.8708 ± 0.0043	0.6671 ± 0.0107	3.33% ± 0.41%	0.6090 ± 0.0079
	Swin-S	0.8636 ± 0.0036	0.7392 ± 0.0054	3.75% ± 0.37%	0.6576 ± 0.0045
CIFAR100	ResNet-50	0.5632 ± 0.0032	0.9431 ± 0.0005	23.75% ± 2.09%	0.8524 ± 0.0009
	Swin-T	0.6001 ± 0.0033	0.9756 ± 0.0003	28.52% ± 1.54%	0.9112 ± 0.0010
	ResNet-101	0.5654 ± 0.0042	0.9379 ± 0.0021	21.25% ± 1.55%	0.8484 ± 0.0036
	Swin-S	0.5900 ± 0.0029	0.9639 ± 0.0006	31.02% ± 1.74%	0.8978 ± 0.0014

The privacy leakage varies on different models and datasets. Similar to [60, 62], we analyze the overfitting levels of victims models in Figure 4a. The overfitting level indicates the accuracy difference of a model between its training and inference. Figure 4a illustrates the results of Transformers and CNNs in CIFAR10 and CIFAR100. We conclude that a more overfitted model comes with higher membership inference attack accuracy. More importantly, at the same overfitting level, Transformers always suffer from higher attack accuracy.

In conclusion, the results strongly suggest that Transformers are more susceptible to network-based and likelihood-based membership inference attacks compared to CNNs. Transformers consistently demonstrate higher vulnerability to membership inference attacks across various metrics.



(a) ResNet-50 vs Swin-T on CIFAR10

(b) ResNet-101 vs Swin-S on CIFAR10

(c) ResNet-50 vs Swin-T on CIFAR100

(d) ResNet-101 vs Swin-S on CIFAR100

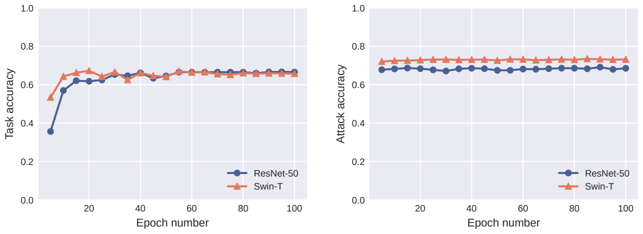
Figure 2: ROC curves for membership inference attacks on CIFAR10 and CIFAR100. Comparisons between CNNs and Transformers.

Table 4: Results of attribute inference attacks on CelebA.

	Task acc ↑	Attack acc ↑	Macro F1 ↑
ResNet-50	0.6666 ± 0.0020	0.6854 ± 0.0015	0.3753 ± 0.0012
Swin-T	0.6587 ± 0.0023	0.7312 ± 0.0014	0.5530 ± 0.0019
ResNet-101	0.6431 ± 0.0029	0.6291 ± 0.0023	0.4262 ± 0.0009
Swin-S	0.6569 ± 0.0024	0.7369 ± 0.0036	0.5536 ± 0.0015

4.3 Evaluation on Attribute Inference Attacks

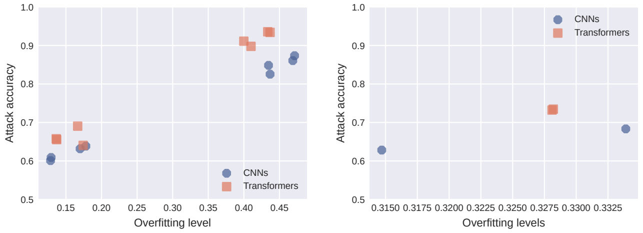
Table 4 presents the results of attribute inference attacks on CelebA. Similarly to membership inference attacks, we categorize CNN and Transformer models into two groups based on their parameter sizes. The table shows that within each group, CNNs and Transformers achieve similar task accuracy scores, approximately 0.65. However, when considering the attack accuracy and Macro F1 score, Transformers consistently outperform CNNs. The results from attribute inference attacks align with our previous findings from membership inference attacks, emphasizing the increased vulnerability of Transformer models to privacy attacks.



(a) Performance of victim models on CelebA

(b) Performance of privacy attack on CelebA

Figure 3: The performance of attribute inference attacks against ResNet-50 and Swin-T on CelebA under different numbers of epochs.



(a) Membership inference

(b) Attribute inference

Figure 4: The performance of privacy attacks against both CNNs and Transformers with various models and datasets under different overfitting levels.

Figure 3 presents the performance of task accuracy and attack accuracy in attribute inference attacks using ResNet-50 and Swin-T on CelebA over 100 epochs. Figure 3a shows that although ResNet-50 and Swin-T models start with different task accuracy scores, both models gradually converge to similar task accuracy scores of around 0.65. However, when we examine the attack accuracy in Figure 3b, the attack accuracy on Swin-T consistently outperforms that on ResNet-50 throughout the 100 epochs. This reveals that Transformers like Swin-T are more vulnerable to attribute inference attacks than ResNet-50 from the start of the attack training to the end.

We further analyze the relationship between the attack performance and the overfitting levels of victim models in Figure 4b. We have made a similar discovery to the previous evaluation: Transformers suffer from higher attack accuracy than CNNs when the victim models are at the same overfitting level.

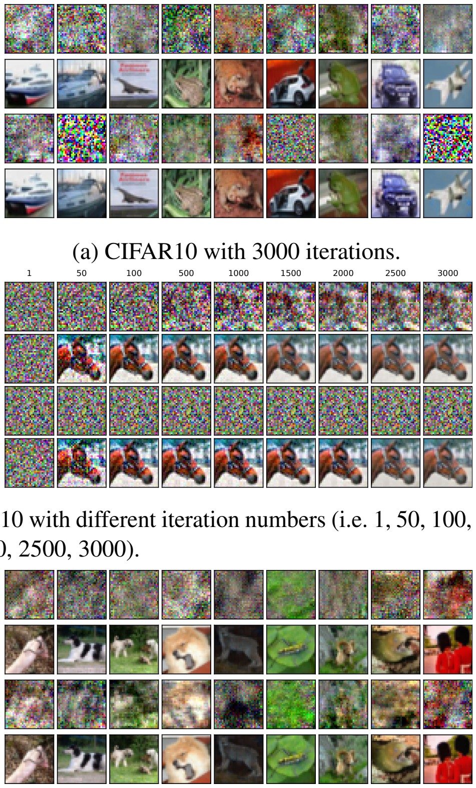
4.4 Evaluation on Gradient Inversion Attacks

Table 5 presents the results of gradient inversion attacks on CNNs and Transformers using CIFAR10. Similar to our previous evaluations, we still compare these two model architectures in groups. The attacks are evaluated by multiple metrics, which measure reconstruction results between ground truth images and reconstruction images. The table clearly shows that the attacks on Transformers outperform the attacks on CNNs by a significant margin. Figure 5a provides examples of the reconstruction results. The attacks on CNNs fail to generate high-quality reconstruction images, whereas the attacks on Transformers produce remarkably accurate reconstructions that closely resemble the originals.

Figure 5b provides the reconstruction results of the gradi-

Table 5: The results of gradient inversion attacks on CNNs and Transformers on CIFAR10.

	MSE ↓	PSNR ↑	LPIPS ↓	SSIM ↑
ResNet-50	1.3308 ± 0.6507	11.30 ± 2.24	0.1143 ± 0.0403	0.0946 ± 0.0989
Swin-T	0.0069 ± 0.0071	36.24 ± 5.21	0.0012 ± 0.0016	0.9892 ± 0.0118
ResNet-101	1.2557 ± 0.6829	11.58 ± 2.16	0.1461 ± 0.1012	0.0784 ± 0.0675
Swin-S	0.0063 ± 0.0083	37.85 ± 6.15	0.0016 ± 0.0028	0.9878 ± 0.0128


(a) CIFAR10 with 3000 iterations.

(b) CIFAR10 with different iteration numbers (i.e. 1, 50, 100, 500, 1000, 1500, 2000, 2500, 3000).

(c) ImageNet1K with 3000 iterations.

Figure 5: The performance of gradient inversion attacks on CNNs and Transformers. From the top row to the bottom in each subfigure are ResNet-50, Swin-T, ResNet-101, and Swin-S.

ent inversion attacks over multiple iterations. It demonstrates the transformation of a raw dummy image towards a reconstruction that closely resembles the original image as the attack training continues. The reconstruction results reveal the varying degrees of success achieved by the attacks on different models. In the case of ResNet-50, the reconstructed image shows limited resemblance to the original image. When attacking ResNet-101, the reconstruction result fails to capture any meaningful information. On the other hand, when targeting Transformers such as Swin-T and Swin-S, the attacks yield highly accurate reconstruction results since early iterations.

In the evaluation of gradient inversion attacks on ImageNet1K, we still compare the performance of ResNet-50, ResNet-101, Swin-T, and Swin-S models. Randomly selected images from ImageNet1K are used to generate reconstruction results, shown in Figure 5c. Similar to the previous experiments on CIFAR10, the attacks on ResNet variants have limited success in reconstructing the original images. In contrast, the attacks on Transformer models (Swin-T and Swin-S) yield significantly better reconstruction results.

These findings highlight the higher vulnerability of Transformers to gradient inversion attacks compared to CNNs. This raises questions about specific architectural features in Transformers that contribute to their increased vulnerability to privacy attacks. We further analyze the vulnerability of architectural features in Section 5.

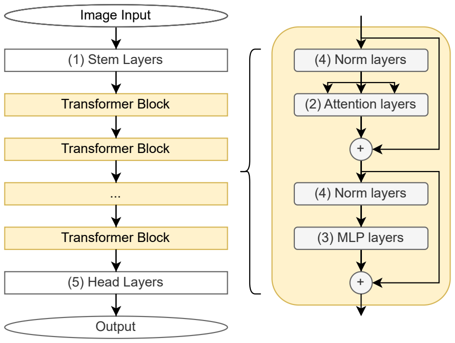
5 Which Architectural Features Can Lead to Higher Privacy Leakage?

In this section, we answer the fourth research question (**RQ4**). We delve into analyzing architectural features that can potentially lead to privacy leakage. Through a comprehensive analysis, we first examine the impact of partitioning a transformer model on privacy vulnerabilities. Then, we investigate the influence of various micro designs on model privacy and conduct an ablation study on these micro designs.

5.1 Segmenting a Transformer Model to Analyze the Privacy Leakage

One of the key distinctions between a CNN and a Transformer lies in the design of their respective blocks. In this section, we focus on segmenting a Transformer-based model and assessing the potential influence of specific layers on privacy leakage. As there are only prediction results and model representations from membership inference attacks and attribute inference attacks, it is difficult to evaluate some specific layers using these attacks. However, gradient inversion attacks offer a more precise means of evaluation, as they require a comprehensive list of gradients from each layer of the victim model.

In our analysis, we employ ViT-B as the victim model. This is because ViT models are one of the first Transformers in computer vision. Additionally, ViT models incorporate several common modules shared across various Transformer architectures. We utilize gradient inversion attacks as our chosen attack method. Rather than supplying all the gradients to the attacks, we selectively provide specific gradients for evaluation purposes. The architecture of a Vision Transformer (an example of ViT-B) is depicted in Figure 6. To facilitate our assessment, we divide the model into five distinct modules based on their layer designs. Subsequently, we evaluate the influence of gradients obtained from each module individually.



(a) Transformer Architecture

(b) Transformer Block

Figure 6: An illustration of a Vision Transformer architecture and the modules we focus on evaluating. Numbers (1) - (5) denote the modules we evaluate in gradient inversion attacks. (a): The whole model architecture. (b): Detailed architecture of each Transformer block.

Table 6: The performance of gradient inversion attacks when segmenting ViT-B to make a selection of gradients.

Layers	Num of layers	Params	MSE ↓	PSNR ↑
All	152	85.65M	0.0007 ± 0.0003	43.70 ± 1.84
Stem	4	0.59M	0.0000 ± 0.0000	67.43 ± 5.03
Attention	48	28.34M	0.0020 ± 0.0009	39.61 ± 2.76
MLP	48	56.66M	0.0036 ± 0.0016	36.98 ± 2.59
Norm	48	0.05M	0.0040 ± 0.0018	36.57 ± 2.56
Head	4	0.01M	0.2776 ± 0.2312	19.01 ± 3.89

If the attack using gradients from Module A yields a higher attack accuracy compared to the attack using gradients from Module B, it suggests that Module A is more likely to reveal more information about the data samples than Module B.

- Module 1: Stem layers. This module receives the model's input and has patch embedding and position embedding layers.
- Module 2: Attention layers. They are in the Transformer block, and this module is the main reason why Transformers are different from CNNs.
- Module 3: MLP layers. They are also in the Transformer block.
- Module 4: Norm layers. They are located right before the attention layers and MLP layers. LayerNorm is often used as Norm layers in Transformers.
- Module 5: Head layers. They are the last few layers for producing the output of the model. A few fully connected layers could be used as head layers.

Table 6 presents the reconstruction results of gradient inversion attacks when only gradients from selected layers are utilized in the attacks. The "All" layers represent the default attack scenario, where gradients from all layers are employed. The stem layers contain the patch embedding and position embedding processes. These layers exhibit minimal changes in the output compared to the original image sample. As the stem layers comprise only four layers, they can be relatively easier to attack compared to other types of layers. Consequently, the attacks on the stem layers demonstrate excellent performance, and we will further assess stem layers in the next subsection.

Table 7: Attack performance on ConvNeXt-T compared with ResNet-50 and Swin-T. NN MIA for Network-based membership inference, AIA for attribute inference, and GIA for gradient inversion.

		ResNet-50	ConvNeXt-T	Swin-T
NN MIA	Attack acc ↑	0.6385 ± 0.0078	0.7471 ± 0.0052	0.6904 ± 0.0052
AIA	Attack acc ↑	0.6854 ± 0.0015	0.7203 ± 0.0020	0.7312 ± 0.0014
	Macro F1 ↑	0.3753 ± 0.0012	0.5469 ± 0.0011	0.5530 ± 0.0019
GIA	MSE ↓	1.5096 ± 0.5538	0.0177 ± 0.0171	0.0069 ± 0.0071
	PSNR ↑	10.58 ± 1.87	31.88 ± 5.04	36.24 ± 5.21
	LPIPS ↓	0.1624 ± 0.0613	0.0032 ± 0.0055	0.0012 ± 0.0016
	SSIM ↑	0.0896 ± 0.0544	0.9666 ± 0.0451	0.9892 ± 0.0118

Among the attacks conducted with the remaining selected layers, the attack that demonstrates the best performance is the one utilizing "attention layers." This attack achieves an MSE of 0.0020 and a PSNR of 39.61. These results indicate that the attention layers are more susceptible to attacks, suggesting that they potentially leak more information about the data samples.

5.2 Impact of Other Micro Designs on Privacy

In the previous subsection, we established that attention layers within Transformers can contribute to privacy leakage. However, it is important to recognize that other micro designs within Transformers may also impact privacy vulnerabilities. One example is ConvNeXt [46], a convolutional neural network that incorporates multiple schemes from Transformers, similar to the Swin Transformer. ConvNeXt-T, ResNet-50, and Swin-T all share a similar parameter size, with ConvNeXt-T having approximately 27.83 million parameters. This allows for a direct comparison of the attack performance between ConvNeXt-T and the other two models. When ConvNeXt-T is tested on CIFAR10 using the same attack settings, it achieves a task accuracy of 0.8258. This indicates that we can compare the attack performance of ConvNeXt-T with ResNet-50 and Swin-T. The results presented in Table 7 further confirm the high attack performance on ConvNeXt-T. These findings suggest that ConvNeXt-T and Swin-T exhibit vulnerability to various attacks at a similar level.

The difference between ConvNeXt and the Swin Transformer lies in the attention module. This allows us to explore and investigate the privacy implications of various micro designs in model architectures beyond just the attention layers. By studying the impact of different micro designs, we can gain deeper insights into the specific architectural features that may pose privacy risks.

We continue with utilizing gradient inversion attacks as a tool for further examination. Following the design process outlined in [46], we have made several adjustments based on our own analysis. Since ConvNeXt is constructed incrementally from ResNet, we meticulously scrutinized each model architecture to investigate the steps that contribute most significantly to privacy leakage. We focus on ResNet-50 and ConvNeXt-T models and conduct tests on a total of 14 model architectures using randomly selected samples from CIFAR10.

The analysis involves a 14-step process, where we begin by modifying the overall architecture (Steps 1 to 4) based on ResNet-50. Subsequently, we align the bottleneck design (Steps 5 to 7) and refine the stem layer design in Step 8. Finally, we align the micro designs (Steps 9 to 14) to achieve ConvNeXt-T. This 14-step process, from macro to micro levels, as outlined in [46], has been established as an optimal procedure for designing ConvNeXt. Leveraging this proven methodology, we utilize it to analyze the impact on privacy. By identifying the critical steps that contribute to privacy leakage, we can assess the influence of micro designs on the privacy of model architectures. This analysis provides valuable insights into understanding the specific stages or modifications that may pose risks to privacy. The 14 steps are outlined below:

1. **ResNet-50:** We begin our process with this model.
2. **Changing channel dimensions:** Each stage in ResNet-50 uses different channel dimensions (i.e., $(64, 128, 256, 512)$). To align with ConvNeXt-T, we modify these dimensions to $(96, 192, 384, 768)$.
3. **Changing the stage compute ratio:** ResNet employs a multi-stage design that modifies channel dimensions. ResNet-50 has a stage compute ratio of $(3, 4, 6, 3)$, while ConvNeXt and Swin Transformers adopt $(3, 3, 9, 3)$. We follow this adjustment in this step.
4. **Applying "Patchify":** Vision transformers process input images by sliding them into patches. Here, we replace the stem convolutional layers with a kernel size of (4×4) and a stride of 4.
5. **Applying "ResNeXtify":** ResNeXt [79] introduced grouped convolution, reducing parameter size while maintaining performance. We utilize depth-wise convolution, which employs the same number of groups and channels.
6. **Using the inverted bottleneck:** The inverted bottleneck design is widely employed in models like MobileNet, ConvNeXt, and Swin Transformers. We incorporate this step into our process.
7. **Enlarging kernel sizes:** To align the parameters with ConvNeXt, we adopt a larger kernel size of (7×7) instead of (3×3).
8. **Forming the new stem layers:** In this step, we remove the activation and maxpool layers, originally part of ResNet.
9. **Changing ReLU to GELU:** The Gaussian Error Linear Unit (GELU) [29] is a variant of ReLU commonly used in Transformers. We introduce this change to the model.
10. **Removing some activation layers:** Transformer blocks typically have fewer activation layers. Here, we retain only one activation layer within the block.
11. **Removing some normalization layers:** We reduce the number of BatchNorm (BN) layers within the block to one.
12. **Changing BN to LN:** Inspired by the prevalent use of LayerNorm (LN) layers [3] in Transformers, we replace the BN layers in our model with LN layers. We also enable bias parameters for all convolutional layers in this step.
13. **Separating downsampling layers:** We move the downsampling layers between stages, introducing an LN layer to ensure stability during training.
14. **Final touches to reach ConvNeXt-T:** We incorporate Stochastic Depth [32] and Layer Scale [72] in the final stage to complete the ConvNeXt-T model.

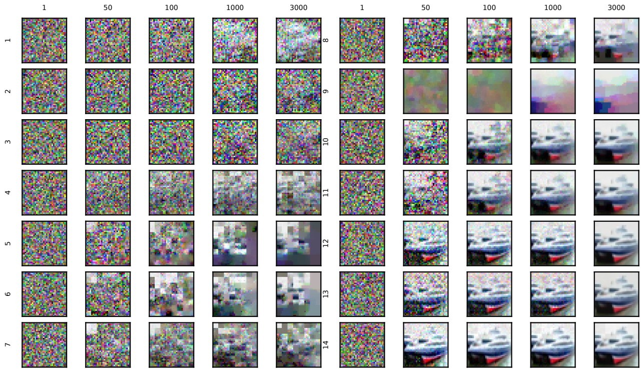


Figure 7: The performance of gradient inversion attacks on each architecture changing from ResNet-50 to ConvNeXt-T with several selected iterations (i.e., 1, 50, 100, 1000, 3000). Model architectures from 14 steps are shown.

Please refer to [86] for more detailed architecture specifications.

Figure 7 and Table 8 show the performance of gradient inversion attacks on each model architecture. The 14-step changes from macro to micro levels exhibit fluctuations but generally show a trend of improved attack performance.

Figure 7 provides qualitative results of the attacks. During the initial stages (Steps 1 to 3), the attacks struggle to reconstruct proper images. In the middle stages, some information emerges in the reconstruction results, but to a limited extent. The reconstruction results improve in the later stages (After Step 10). At last, using ConvNeXt-T, which is step 14, the attacks achieve a good attack performance.

Table 8: The results of gradient inversion attacks on model architectures from 14 steps. Some significant changes in results are marked in bold.

Steps	Task acc ↑	MSE ↓	PSNR ↑	LPIPS ↓	**SSIM** ↑
1. ResNet-50	0.8220 ± 0.0039	1.5096 ± 0.5538	10.58 ± 1.87	0.1624 ± 0.0613	0.0896 ± 0.0544
2. Channel dim	0.8240 ± 0.0072	1.4706 ± 0.5710	10.74 ± 1.97	0.1724 ± 0.0616	0.0826 ± 0.0405
3. Stage ratio	0.8282 ± 0.0040	1.5286 ± 0.5246	10.56 ± 2.05	0.1834 ± 0.0581	0.0731 ± 0.0613
4. Patchify	0.8293 ± 0.0061	**0.9011** ± **0.4376**	**12.97** ± **2.10**	**0.0867** ± **0.0436**	**0.1727** ± **0.0794**
5. ResNeXtify	0.8397 ± 0.0033	1.2415 ± 0.6934	11.86 ± 2.77	0.1066 ± 0.0391	0.1334 ± 0.0950
6. Inv bottleneck	0.8407 ± 0.0058	1.1123 ± 0.4994	12.06 ± 2.19	0.0989 ± 0.0290	0.1429 ± 0.0844
7. Kernel sizes	0.8432 ± 0.0052	0.8206 ± 0.3543	13.40 ± 2.30	0.0821 ± 0.0355	0.2353 ± 0.0766
8. New stem	0.8459 ± 0.0043	0.5684 ± 0.3564	15.43 ± 3.01	0.0752 ± 0.0381	0.4924 ± 0.1205
9. ReLU to GELU	0.8436 ± 0.0027	1.0540 ± 0.5075	12.42 ± 2.61	0.2422 ± 0.0904	0.1746 ± 0.1166
10. Removing Act	0.8480 ± 0.0064	**0.0215** ± **0.0150**	**29.93** ± **3.58**	**0.0049** ± **0.0026**	**0.9562** ± **0.0224**
11. Removing BN	0.8491 ± 0.0059	0.0198 ± 0.0139	30.57 ± 4.12	0.0045 ± 0.0032	0.9605 ± 0.0232
12. BN to LN	0.8501 ± 0.0031	**0.0049** ± **0.0044**	**36.86** ± **3.96**	**0.0005** ± **0.0003**	**0.9927** ± **0.0064**
13. Sep downsamp	0.8553 ± 0.0070	0.0121 ± 0.0171	33.79 ± 4.69	0.0011 ± 0.0008	0.9859 ± 0.0151
14. ConvNeXt	0.8523 ± 0.0064	0.0177 ± 0.0171	31.88 ± 5.04	0.0032 ± 0.0055	0.9666 ± 0.0451

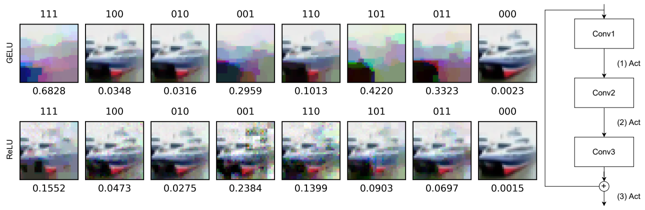
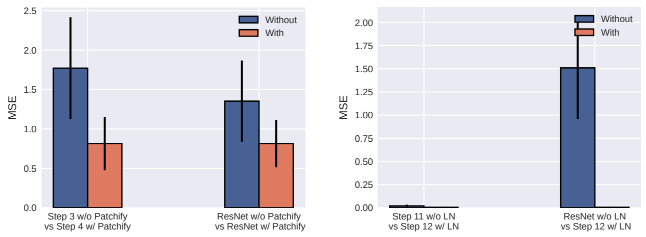


Figure 8: The performance of gradient inversion attacks on models with different positions of activation layers. The position of three activation layers is illustrated on the right, with two between convolutional layers and one after the addition operation. The three digits on the top of the subfigures show whether the activation layer on this position is added or not. The bottom of each subfigure provides MSE values for the attack on this model (i.e., models with GELU or ReLU).



(a) Patchify

(b) LN

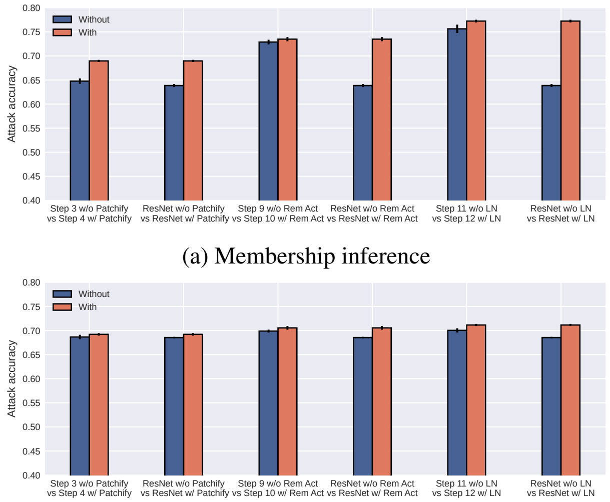
Figure 9: The performance of gradient inversion attacks on model architectures with or without some micro designs.

Table 8 presents more information on gradient inversion attacks, highlighting their performance in various architectural changes. Several steps exhibit significant changes compared to other steps. We define a step as a significant change when a metric notably decreases or increases. To illustrate, we consider the MSE metric, which we define a step as a significant change if it experiences a reduction exceeding 50%. Our analysis reveals three significant increases in attack performance, each correlated with specific architectural changes: The first one occurs when applying "Patchify" to stem layers; The second one occurs with the removal of some activation layers; The third one happens when changing BN to LN. These findings underscore the pivotal role these specific architectural changes play in determining the model's vulnerability to privacy attacks.

5.3 Ablation Study on Micro Designs

Ablation study on the design of activation layers. One of the differences between a Transformer block and a ResNet block is that a Transformer block has fewer activation layers. Leaving fewer activation layers in the model boosts the attack performance. As illustrated in Figure 8, when different activation layers are removed, there is a noticeable improvement in the attack performance. Particularly, the removal of the third activation layer, located after the skip connection of the ResNet block, results in a significant enhancement in attack accuracy. This observation suggests that this activation layer introduces a non-linear process that reduces the amount of information available for the attack, thus making it harder to reconstruct the original input. By removing this layer, the attack performance is improved. The analysis further highlights that changing ReLU to GELU results in a marginal reduction in attack performance. This is attributed to the fact that GELU is a smoother approximation of ReLU, which strengthens the model's robustness and generalization [4]. Using GELU makes the adversary extract less information from data samples, consequently decreasing their attack efficacy. In sum, the most significant enhancement in attack performance occurs when all activation layers are removed from the model architecture.

Ablation study on the design of stem layers and LN layers. The additional analysis presented in Figure 9 highlights two more features that appear to have an impact on model privacy leakage. Firstly, Figure 9a demonstrates that the utilization of "Patchify" reduces reconstruction MSE results. As



(a) Membership inference

(b) Attribute inference

Figure 10: The performance of privacy attacks on model architectures with or without some features.

this process involves modifications to the stem layers, we believe that these stem layers are crucial for privacy attacks. Secondly, Figure 9b illustrates that changing BN to LN results in improved reconstruction results. This suggests that incorporating LN also contributes to the model's privacy leakage. These findings emphasize the significance of stem layers and LN layers on model privacy leakage.

Ablation study on micro designs for membership inference attacks and attribute inference attacks. Figure 10 shows the impact of three micro designs on more attacks. Just like previous studies, adding these designs can also increase privacy leakage in these two attacks.

In summary, attention modules, as well as the design of activation layers, stem layers, and LN layers, are the key architectural features that lead to more privacy leakage. We provide more discussions in Section 6.

6 Discussion

In the previous sections, we discovered that four design components in Transformers could result in privacy leakage: attention modules, activation layers, stem layers, and LN layers. In this section, we would like to provide a more in-depth discussion of these modules.

6.1 The Impact of Attention Modules

The receptive field of a model refers to the information received within a specified range by a neuron in a model layer. In a fully connected neural network, each neuron receives input from the elements of the entire input sample. Due to the convolution operation, the neuron in a convolutional network receives input limited to its receptive field. The range of the receptive field is defined by the convolution templates in CNNs. This design allows CNNs to capture local patterns in the input data efficiently. The receptive field has a theoretical limit. Some researchers have demonstrated that the effective receptive field (i.e., the receptive field's effective area) is smaller than the theoretical receptive field [50]. From a privacy perspective, a CNN model could only reveal part of sensitive information from the input sample due to the design of localized convolution templates.

Transformers employ the multi-head self-attention mechanism, also known as attention modules. The input sample is taken as a sequence of flattened 2D patches. The attention module receives the input sequence and generates its representation of the sequence by mapping the query and the key-value pairs to the output. Transformers tend to have much larger receptive fields than CNNs due to the fact that their attention module is computed with the entire input sequence [16,74]. In terms of privacy, a Transformer model is able to extract more sensitive information than a CNN model because of its wide-angle receptive field. Hence, Transformers are more prone to attacks than CNNs, as demonstrated by our evaluation based on three popular privacy attacks.

6.2 The Impact of Micro Designs

There are micro-design components with the potential to leak sensitive data from input samples. Activation layers such as ReLU and GELU add a layer of complexity to the model by making it a non-linear function. Removing some activation layers simplifies the logic of attacks. Stem layers receive an input sample and perform some preliminary processing. As the representation after the stem layers remains similar to the original input image, there is a high possibility of extracting private information because of the design of the stem layers. Changing BN to LN is also likely to aid the attack process and allow the adversary to achieve higher attack performance.

6.3 The Impact of Overfitting

Previous work claimed that privacy attacks are caused mainly by the undesirable overfitting issue in deep learning models [44, 62]. Overfitting normally occurs when a model performs well on the training data, but poorly on the validation data. The overfitting issue tends to become severe on an over-trained model with a large number of parameters. Deep learning models are exposed to privacy threats due to the overfitting effect. In our work, we find that model architectures have impacts on the performance of privacy attacks, which can *not* be attributed solely to the overfitting effect. Indeed, our experiments validate that the variation in performance is due to the difference in model architectures. For models with the same level of parameter sizes, Transformers tend to be more vulnerable to privacy attacks than CNNs. More importantly, for models with the same overfitting level, our conclusion still holds that Transformers are more vulnerable to privacy attacks than CNNs. We have then identified some architectural features that could be responsible for privacy leakage.

6.4 Potential (Incorrect) Explanations for the Vulnerability of Different Models Against Privacy Attacks

Here, we explore several potential explanations for our experimental results and shed more insights on our conclusion.

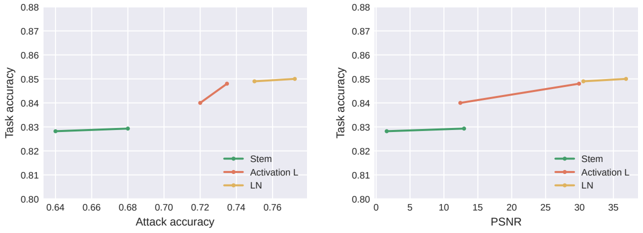
- The attacks on Transformers are more effective than those on CNNs. Is it due to random noise? To dismiss this concern, we have conducted multiple runs for each experiment setting and calculated the mean and standard deviation scores for our results. In total, We have done around 100 different experiment settings, involving 1400 experiments, 1700 training models, and 1200 training hours, which averages out the impact of random noise.
- Is it due to the overfitting of the victim models? To account for this, we meticulously compare the attacks on CNNs and Transformers when their victim models are trained on the same level of overfitting. By doing so, we minimize the factor of overfitting for a fair assessment.
- Is it due to the immature training of the victim models? In order to conduct our experiments, it was necessary to split the dataset into multiple subsets for both victim and shadow models. Consequently, it is reasonable to expect relatively low accuracy on the victim models for CIFAR10, CIFAR100, and CelebA (compared to the models trained on a full dataset). This aligns with the findings of the literature on privacy attacks, which have reported similar results [7, 27, 44].
- Is it due to other training factors? We ensure that the training recipes for each victim model remain consistent across all comparisons. Consequently, we can confidently exclude other training factors from the comparison study when drawing our conclusion.

After eliminating these potential explanations, we believe that our experimental results should be best explained by several architectural features mentioned in Section 5. Please see Appendix B for more theoretical discussions.

7 How to Exploit the Privacy Impact of Model Components?

7.1 Modifying Model Components as a Defense Mechanism

Based on our discoveries in previous sections, we propose to explore the modification of model components as a defense mechanism against privacy attacks. We aim to achieve a tradeoff between utility and privacy. In Figure 11, we demonstrate the influence of certain micro-design changes on the efficacy of privacy attacks. Notably, we observe that the task accuracy is minimally impacted in each test, indicating that the proposed defense measures do not significantly compromise the model's overall performance. However, we can observe a decreased effectiveness of membership inference attacks and gradient inversion attacks when these model components are not applied.



(a) Membership inference (b) Gradient inversion

Figure 11: The impact of the performance of privacy attacks when model components change. In each line, the left and right points represent the results without and with the micro design modifications, respectively.

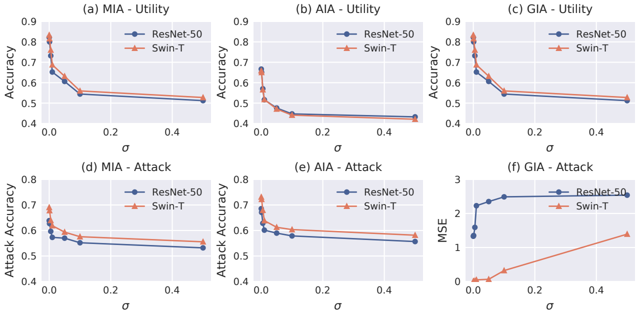


Figure 12: The performance of privacy attacks when DPSGD is applied. The utility performance of victim models and the attack performances are given when various DP noises are given.

These findings suggest that the proposed micro-design modifications can serve as effective countermeasures against privacy attacks, while the task accuracy remains almost intact.

7.2 Adding Perturbations as a Defense Mechanism

The privacy leakage issue observed in Transformers highlights the need for an enhanced privacy treatment compared to CNNs. Specifically, when employing perturbation as a defense mechanism, such as incorporating differential privacy noises into the model parameters, it is better to increase the level of noise specifically for Transformers and Transformer-

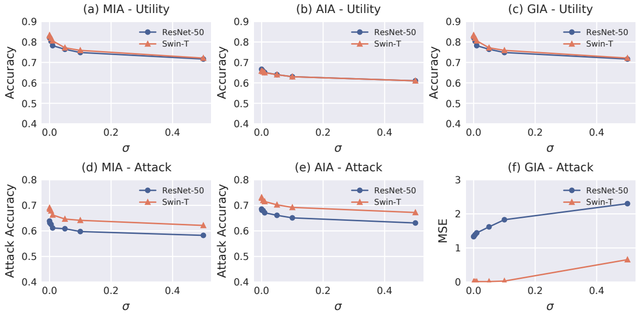


Figure 13: The performance of privacy attacks when DPSGD is applied in stem layers. The utility performance of victim models and the attack performances are given when various DP noises are given.

like models. Figure 12 illustrates the outcomes of three privacy attacks when DPSGD [1] is applied. As the differential privacy noise increases, a noticeable decline in the utility performance of victim models is observed (depicted in subfigures (a) to (c)). Concurrently, subfigures (d) to (f) reveal diminishing attack performances, with Transformers (Swin-T) consistently exhibiting superior attack performance at equivalent noise levels. From a defensive standpoint, adding more noise to Transformers is recommended.

Another approach to consider is a layer-wise perturbation defense mechanism, where noises are added to selected layers only. In this scenario, it would be interesting to introduce additional noises to the layers that are more susceptible to privacy leakages, such as activation layers, stem layers, LN layers, and attention modules. Paying special attention to these "privacy-leakage" layers could also achieve satisfactory privacy protection. Figure 13 demonstrates the impact of privacy attacks when DPSGD is applied to stem layers. As the differential privacy noise escalates, it influences both utility and attack performance, demonstrating the efficacy of adding noises to targeted layers.

8 Conclusion

In this study, we pioneer the exploration of privacy risks on model architectures, especially CNNs and Transformers. We have conducted a comparison of three prominent privacy attacks, i.e., membership inference attacks, attribute inference attacks, and gradient inversion attacks. We discover that Transformers tend to be more vulnerable to privacy attacks than CNNs. As a result, many Transformers-inspired CNN designs, such as ConvNeXt, are also susceptible to privacy threats. A number of Transformers' features, including the design of activation layers, the design of stem layers, the design of LN layers, and the attention modules, are likely to incur high privacy risks.

It is still challenging to establish accurate and theoretical explanations for why certain architectural features are critical to privacy preservation. We believe that these analyses require further experimental campaigns, and we intend to study this matter in our future work.

Acknowledgments

We thank all anonymous reviewers and our shepherd for their constructive comments. This work is funded in part by ARC Discovery Grant (DP230100246) and ARC Linkage Grant (LP220200808).

References

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, and et al. Deep Learning with Differential Privacy. In *CCS*, 2016.

[2] André Araujo, Wade Norris, and Jack Sim. Computing receptive fields of convolutional neural networks. *Distill*, 2019.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[4] Yutong Bai, Jieru Mei, Alan L Yuille, and Cihang Xie. Are Transformers more robust than CNNs? In *NeurIPS*, 2021.

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, and et al. Language Models are Few-Shot Learners. In *NeurIPS*, 2020.

[6] Sebastien Bubeck, Ronen Eldan, Yin Tat Lee, and Dan Mikulincer. Network size and size of the weights in memorization with two-layers neural networks. In *NeurIPS*, 2020.

[7] Nicholas Carlini, Steve Chien, Milad Nasr, Shuang Song, Andreas Terzis, and Florian Tramèr. Membership Inference Attacks From First Principles. In *S&P*, 2022.

[8] Nicholas Carlini, Matthew Jagielski, Chiyuan Zhang, Nicolas Papernot, Andreas Terzis, and Florian Tramer. The Privacy Onion Effect: Memorization is Relative. In *NeurIPS*, 2022.

[9] Dingfan Chen, Ning Yu, Yang Zhang, and Mario Fritz. GAN-Leaks: A Taxonomy of Membership Inference Attacks against Generative Models. In *CCS*, 2020.

[10] Min Chen, Zhikun Zhang, Tianhao Wang, Michael Backes, Mathias Humbert, and Yang Zhang. When Machine Unlearning Jeopardizes Privacy. In *CCS*, 2021.

[11] Christopher A. Choquette-Choo, Florian Tramer, Nicholas Carlini, and Nicolas Papernot. Label-Only Membership Inference Attacks. In *ICML*, 2021.

[12] Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. Capacity and Trainability in Recurrent Neural Networks. In *ICLR*, 2017.

[13] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *CVPR*, 2009.

[14] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*, 2019.

[15] Mingyu Ding, Bin Xiao, Noel Codella, Ping Luo, Jingdong Wang, and Lu Yuan. Davit: Dual attention vision transformers. In *ECCV*, 2022.

[16] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, and et al. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In *ICLR*, 2021.

[17] Simon S. Du, Jason D. Lee, and Yuandong Tian. When is a Convolutional Filter Easy to Learn? In *ICLR*, 2018.

[18] Vasisht Duddu and Antoine Boutet. Inferring Sensitive Attributes from Model Explanations. In *CIKM*, 2022.

[19] Vitaly Feldman. Does learning require memorization? a short tale about a long tail. In *STOC*, 2020.

[20] Matt Fredrikson, Somesh Jha, and Thomas Ristenpart. Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures. In *CCS*, 2015.

[21] Matthew Fredrikson, Eric Lantz, Somesh Jha, Simon Lin, David Page, and et al. Privacy in Pharmacogenetics: An End-to-End Case Study of Personalized Warfarin Dosing. In *USENIX Security*, 2014.

[22] Jonas Geiping, Hartmut Bauermeister, Hannah Dröge, and Michael Moeller. Inverting Gradients - How easy is it to break privacy in federated learning? *NeurIPS*, 2020.

[23] Surbhi Goel, Adam Klivans, and Raghu Meka. Learning One Convolutional Layer with Overlapping Patches. In *ICML*, 2018.

[24] Ali Hatamizadeh, Hongxu Yin, Holger R. Roth, Wenqi Li, and et al. GradViT: Gradient Inversion of Vision Transformers. In *CVPR*, 2022.

[25] Jamie Hayes, Luca Melis, George Danezis, and Emiliano De Cristofaro. LOGAN: Membership Inference Attacks Against Generative Models. *PoPETs*, 2019.

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

[27] Xinlei He and Yang Zhang. Quantifying and Mitigating Privacy Risks of Contrastive Learning. In *CCS*, 2021.

[28] Yang He, Shadi Rahimian, Bernt Schiele, and Mario Fritz. Segmentations-Leak: Membership Inference Attacks and Defenses in Semantic Image Segmentation. In *ECCV*, 2020.

[29] Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

[30] Andrew Howard, Mark Sandler, Bo Chen, Weijun Wang, Liang-Chieh Chen, and et al. Searching for MobileNetV3. In *ICCV*, 2019.

[31] Hongsheng Hu, Zoran Salcic, Lichao Sun, Gillian Dobbie, Philip S. Yu, and Xuyun Zhang. Membership Inference Attacks on Machine Learning: A Survey. *ACM Comput. Surv.*, 2022.

[32] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q. Weinberger. Deep Networks with Stochastic Depth. In *ECCV*, 2016.

[33] Bargav Jayaraman and David Evans. Are Attribute Inference Attacks Just Imputation? In *CCS*, pages 1569–1582, 2022.

[34] Bargav Jayaraman, Lingxiao Wang, Katherine Knipmeyer, Quanquan Gu, and David Evans. Revisiting Membership Inference Under Realistic Assumptions. *PoPETs*, 2021.

[35] Yigitcan Kaya and Tudor Dumitras. When Does Data Augmentation Help With Membership Inference Attacks? In *ICML*, 2021.

[36] A. Krizhevsky. Learning multiple layers of features from tiny images. 2009.

[37] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NeurIPS*, 2012.

[38] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, and et al. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1989.

[39] Minhyeok Lee. GELU Activation Function in Deep Learning: A Comprehensive Mathematical Analysis and Performance. *arXiv preprint arXiv:2305.12073*, 2023.

[40] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, and et al. Mvitv2: Improved multiscale vision transformers for classification and detection. In *CVPR*, 2022.

[41] Zheng Li and Yang Zhang. Membership Leakage in Label-Only Exposures. In *CCS*, 2021.

[42] Zhuohang Li, Jiaxin Zhang, Luyang Liu, and Jian Liu. Auditing Privacy Defenses in Federated Learning via Generative Gradient Leakage. In *CVPR*, 2022.

[43] Bo Liu, Ming Ding, Sina Shaham, Wenny Rahayu, Farhad Farokhi, and Zihuai Lin. When Machine Learning Meets Privacy: A Survey and Outlook. *ACM Comput. Surv.*, 2021.

[44] Yugeng Liu, Rui Wen, Xinlei He, Ahmed Salem, Zhikun Zhang, and et al. ML-Doctor: Holistic Risk Assessment of Inference Attacks Against Machine Learning Models. In *USENIX Security*, 2022.

[45] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, and et al. Swin Transformer: Hierarchical Vision Transformer using Shifted Windows. In *ICCV*, 2021.

[46] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, and et al. A ConvNet for the 2020s. In *CVPR*, 2022.

[47] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep Learning Face Attributes in the Wild. In *ICCV*, 2015.

[48] Yunhui Long, Lei Wang, Diyue Bu, Vincent Bindschaedler, Xiaofeng Wang, and et al. A Pragmatic Approach to Membership Inferences on Machine Learning Models. In *EuroS&P*, 2020.

[49] Jiahao Lu, Xi Sheryl Zhang, Tianli Zhao, Xiangyu He, and Jian Cheng. APRIL: Finding the Achilles' Heel on Privacy for Vision Transformers. In *CVPR*, 2022.

[50] Wenjie Luo, Yujia Li, Raquel Urtasun, and Richard Zemel. Understanding the Effective Receptive Field in Deep Convolutional Neural Networks. In *NeurIPS*, 2016.

[51] Shagufta Mehnaz, Sayanton V. Dibbo, Ehsanul Kabir, Ninghui Li, and Elisa Bertino. Are Your Sensitive Attributes Private? Novel Model Inversion Attribute Inference Attacks on Classification Models. In *USENIX Security*, 2022.

[52] Luca Melis, Congzheng Song, Emiliano De Cristofaro, and Vitaly Shmatikov. Exploiting Unintended Feature Leakage in Collaborative Learning. In *S&P*, 2019.

[53] Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, and et al. Scalable Extraction of Training Data from (Production) Language Models. *arXiv preprint arXiv:2311.17035*, 2023.

[54] Milad Nasr, Reza Shokri, and Amir Houmansadr. Comprehensive Privacy Analysis of Deep Learning: Passive and Active White-box Inference Attacks against Centralized and Federated Learning. In *S&P*, 2019.

[55] Sayak Paul and Pin-Yu Chen. Vision Transformers Are Robust Learners. *AAAI*, 2022.

[56] Ilija Radosavovic, Raj Prateek Kosaraju, Ross Girshick, Kaiming He, and Piotr Dollar. Designing Network Design Spaces. In *CVPR*, 2020.

[57] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do Vision Transformers See Like Convolutional Neural Networks? In *NeurIPS*, 2021.

[58] Shashank Rajput, Kartik Sreenivasan, Dimitris Papailiopoulos, and Amin Karbasi. An Exponential Improvement on the Memorization Capacity of Deep Threshold Networks. In *NeurIPS*, 2021.

[59] Alexandre Sablayrolles, Matthijs Douze, Cordelia Schmid, Yann Ollivier, and Herve Jegou. White-box vs Black-box: Bayes Optimal Strategies for Membership Inference. In *ICML*, 2019.

[60] Ahmed Salem, Yang Zhang, Mathias Humbert, Pascal Berrang, Mario Fritz, and Michael Backes. ML-Leaks: Model and data independent membership inference attacks and defenses on machine learning models. In *NDSS*, 2019.

[61] Virat Shejwalkar and Amir Houmansadr. Membership Privacy for Machine Learning Models Through Knowledge Transfer. *AAAI*, 2021.

[62] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models. In *S&P*, 2017.

[63] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv preprint arXiv:1409.1556*, 2015.

[64] Congzheng Song and Ananth Raghunathan. Information Leakage in Embedding Models. In *CCS*, 2020.

[65] Congzheng Song and Vitaly Shmatikov. Overlearning Reveals Sensitive Attributes. In *ICLR*, 2020.

[66] Liwei Song and Prateek Mittal. Systematic Evaluation of Privacy Risks of Machine Learning Models. In *USENIX Security*, 2021.

[67] Andreas Peter Steiner, Alexander Kolesnikov, Xiaohua Zhai, Ross Wightman, and et al. How to train your ViT? Data, augmentation, and regularization in vision transformers. *TMLR*, 2022.

[68] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, and et al. Going deeper with convolutions. In *CVPR*, 2015.

[69] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision. In *CVPR*, 2016.

[70] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In *ICML*, 2019.

[71] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In *ICML*, 2021.

[72] Hugo Touvron, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Hervé Jégou. Going deeper with Image Transformers. In *ICCV*, 2021.

[73] Stacey Truex, Ling Liu, Mehmet Emre Gursoy, Lei Yu, and Wenqi Wei. Demystifying Membership Inference Attacks in Machine Learning as a Service. *IEEE Transactions on Services Computing*, 2019.

[74] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, and et al. Attention is All you Need. In *NeurIPS*, 2017.

[75] Zeyu Wang, Yutong Bai, Yuyin Zhou, and Cihang Xie. Can CNNs Be More Robust Than Transformers? In *ICLR*, 2023.

[76] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 2004.

[77] Lauren Watson, Chuan Guo, Graham Cormode, and Alexandre Sablayrolles. On the Importance of Difficulty Calibration in Membership Inference Attacks. In *ICLR*, 2022.

[78] Tete Xiao, Mannat Singh, Eric Mintun, Trevor Darrell, Piotr Dollar, and Ross Girshick. Early Convolutions Help Transformers See Better. In *NeurIPS*, 2021.

[79] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated Residual Transformations for Deep Neural Networks. In *CVPR*, 2017.

[80] Jingjing Xu, Xu Sun, Zhiyuan Zhang, Guangxiang Zhao, and et al. Understanding and Improving Layer Normalization. In *NeurIPS*, 2019.

[81] Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NeurIPS*, 2019.

[82] Jiayuan Ye, Aadyaa Maddi, Sasi Kumar Murakonda, Vincent Bindschaedler, and Reza Shokri. Enhanced Membership Inference Attacks against Machine Learning Models. In *CCS*, 2022.

[83] Samuel Yeom, Irene Giacomelli, Matt Fredrikson, and Somesh Jha. Privacy Risk in Machine Learning: Analyzing the Connection to Overfitting. In *CSF*, 2018.

[84] Hongxu Yin, Arun Mallya, Arash Vahdat, Jose M. Alvarez, Jan Kautz, and Pavlo Molchanov. See Through Gradients: Image Batch Recovery via GradInversion. In *CVPR*, 2021.

[85] Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, and et al. Tokens-to-Token ViT: Training Vision Transformers from Scratch on ImageNet. In *ICCV*, 2021.

[86] Guangsheng Zhang, Bo Liu, Huan Tian, Tianqing Zhu, Ming Ding, and Wanlei Zhou. How Does a Deep Learning Model Architecture Impact Its Privacy? A Comprehensive Study of Privacy Attacks on CNNs and Transformers. *arXiv preprint arXiv:2210.11049*, 2024.

[87] Guangsheng Zhang, Bo Liu, Tianqing Zhu, Ming Ding, and Wanlei Zhou. Label-Only Membership Inference Attacks and Defenses In Semantic Segmentation Models. *IEEE Transactions on Dependable and Secure Computing*, 2022.

[88] Guangsheng Zhang, Bo Liu, Tianqing Zhu, Andi Zhou, and Wanlei Zhou. Visual privacy attacks and defenses in deep learning: A survey. *Artif Intell Rev*, 2022.

[89] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. In *CVPR*, 2018.

[90] Rui Zhang, Song Guo, Junxiao Wang, Xin Xie, and Dacheng Tao. A Survey on Gradient Inversion: Attacks, Defenses and Future Directions. In *IJCAI*, 2022.

[91] Benjamin Zi Hao Zhao, Aviral Agrawal, Catisha Coburn, Hassan Jameel Asghar, and et al. On the (In)Feasibility of Attribute Inference Attacks on Machine Learning Models. In *EuroS&P*, 2021.

[92] Bo Zhao, Konda Reddy Mopuri, and Hakan Bilen. iDLG: Improved Deep Leakage from Gradients. *arXiv preprint arXiv:2001.02610*, 2020.

[93] Ligeng Zhu, Zhijian Liu, and Song Han. Deep Leakage from Gradients. *NeurIPS*, 2019.

[94] Yang Zou, Zhikun Zhang, Michael Backes, and Yang Zhang. Privacy Analysis of Deep Learning in the Wild: Membership Inference Attacks against Transfer Learning. *arXiv preprint arXiv:2009.04872*, 2020.

Table 9: The results of membership inference attacks on more models of CNNs and Transformers on CIFAR10 as a supplement for Section 4.2.

	Task acc ↑	Attack acc ↑
ResNet-50	0.8220 ± 0.0023	0.6385 ± 0.0078
EfficientNet-B4	0.8180 ± 0.0052	0.6115 ± 0.0049
RegNetX-4.0GF	0.8326 ± 0.0018	0.6197 ± 0.0042
Swin-T	0.8335 ± 0.0042	0.6904 ± 0.0052
MViTV2-T	0.8279 ± 0.0057	0.6813 ± 0.0056
DaViT-T	0.8311 ± 0.0045	0.6752 ± 0.0039

Table 10: The results of attribute inference attacks on more models of CNNs and Transformers on CelebA as a supplement for Section 4.3.

	Task acc ↑	Attack acc ↑	Macro F1 ↑
ResNet-50	0.6666 ± 0.0020	0.6854 ± 0.0015	0.3753 ± 0.0012
EfficientNet-B4	0.6192 ± 0.0038	0.6504 ± 0.0031	0.4036 ± 0.0018
RegNetX-4.0GF	0.6248 ± 0.0041	0.6701 ± 0.0016	0.3649 ± 0.0022
Swin-T	0.6587 ± 0.0023	0.7312 ± 0.0014	0.5530 ± 0.0019
MViTV2-T	0.6151 ± 0.0012	0.6908 ± 0.0025	0.4883 ± 0.0024
DaViT-T	0.6517 ± 0.0030	0.7263 ± 0.0015	0.5093 ± 0.0011

A Additional Experimental Results

Tables 9 to 11 show privacy attack results for more models of CNNs and Transformers. The comparison still involves models with similar parameter sizes, and our conclusion remains

Table 11: The results of gradient inversion attacks on more models of CNNs and Transformers on CIFAR10 as a supplement for Section 4.4.

	MSE ↓	PSNR ↑	LPIPS ↓	SSIM ↑
ResNet-50	1.3308 ± 0.6507	11.30 ± 2.24	0.1143 ± 0.0403	0.0946 ± 0.0989
EfficientNet-B4	0.8178 ± 0.7276	14.6970 ± 3.9812	0.1953 ± 0.0816	0.2698 ± 0.1413
RegNetX-4.0GF	0.7937 ± 0.4853	13.8190 ± 2.6519	0.0851 ± 0.0391	0.3197 ± 0.1282
Swin-T	0.0069 ± 0.0071	36.24 ± 5.21	0.0012 ± 0.0016	0.9892 ± 0.0118
MViTV2-T	0.0711 ± 0.0172	23.6450 ± 1.1224	0.0068 ± 0.0033	0.8639 ± 0.0463
DaViT-T	0.0615 ± 0.0425	25.9200 ± 4.7786	0.0090 ± 0.0076	0.9347 ± 0.0489

unchanged that Transformers exhibit higher attack performance.

B More Theoretical Discussion

In Section 6, we offer general discussions on the privacy impact of four components in Transformers. In this section, we delve deeper into more theoretical discussions.

B.1 The Impact of Attention Modules

In Sections 5.1 and 6.1, we show that the utilization of attention modules makes Transformers more susceptible to privacy leakage than CNNs. This susceptibility is attributed to the larger receptive field of attention modules, contributing to increased model memorization and, subsequently, heightened privacy leakage. Let us explore this theoretical framework step by step:

Receptive Field. Given a neural network with n layers, the output of the i-th layer ($i \in \{1,...,n\}$) is the feature map f_i. We have the input image as f_1 and the final output feature map as f_n. We define $\mathcal{R}_i$ as the receptive field size of feature map f_i. In other words, $\mathcal{R}_i$ is related to the process of the i-th layer. The receptive field $\mathcal{R}_i$ can be defined as

$$\mathcal{R}_i = \zeta(\mathcal{R}_{i+1}, \alpha_1, \alpha_2, ..., \alpha_n), \tag{1}$$

where $\alpha_1, \alpha_2, ..., \alpha_n$ are calculation factors in $\mathcal{R}_{i+1}$, and ζ is the calculation process of $\mathcal{R}_{i+1}$.

In a CNN model, the receptive field of a neuron represents the convolution-computation region in the input space that influences the output of that neuron. It signifies the area of the input data to which the neuron reacts. According to [2], the receptive field in a CNN is determined by:

$$\mathcal{R}_i^C = s_{i+1} \cdot \mathcal{R}_{i+1}^C + (k_{i+1} - s_{i+1}), \tag{2}$$

where k_i and s_i are the kernel size and the stride in a convolution layer. Based on Equations (1) and (2), we can define the receptive field size of a neuron in a CNN layer $\mathcal{R}^C$ as follows (we omit i for the simplicity of notations):

$$\mathcal{R}^C = \zeta(k, s), \tag{3}$$

where k and s are the kernel size and the stride.

The receptive field in a Transformer model, utilizing attention mechanisms, significantly differs from CNNs. In Transformers, attention layers calculate each position in the input sequence using query (q), key (k), and value (v) matrices. The output of the attention module is computed as follows:

$$\text{Attention}(q, k, v) = \text{SoftMax}(q \cdot k^T / \sqrt{d}) \cdot v, \tag{4}$$

where d is the scaling factor based on query and key. Based on Equations (1) and (4), we can define the size of the receptive field of an attention layer $\mathcal{R}^T$ as:

$$\mathcal{R}^T = \zeta(q, k, v). \tag{5}$$

As q, k, and v in a Transformer model are derived from the entire input sequence, $\mathcal{R}^T$ tends to be much larger. With Equations (3) and (5), we have:

$$\mathcal{R}^T > \mathcal{R}^C. \tag{6}$$

In summary, Transformers activate a larger receptive field of input data compared to CNNs, which is also supported by empirical research [57].

Model Memorization. The concept of model memorization lacks a formal definition in the research field. Informally, it can be characterized as a label (**y**) memorization for a specific sample **x** if removing this **x** from the training set would change the performance of the model predicting **y** [8, 19]. Other research refers to it as memory capacity, assessing how much information a model can store in its parameters [6, 12, 58]. In either case, model memorization is intricately linked to how the model learns from input data. The relationship between the receptive field and model memorization can be formulated by:

$$\text{increase}(\mathcal{M}) \sim \text{increase}(\mathcal{R}), \tag{7}$$

where $\mathcal{R}$ denotes the receptive field and $\mathcal{M}$ denotes the model memorization. A larger receptive field allows the model to reap more information from the input data, and as Transformers inherently possess a larger receptive field, they exhibit an enhanced ability to absorb more information. With Equations (6) and (7), we have:

$$\mathcal{M}^T > \mathcal{M}^C, \tag{8}$$

where $\mathcal{M}^T$ is the model memorization in Transformers, and $\mathcal{M}^C$ is the model memorization in CNNs.

Privacy Leakage. Model memorization introduces potential privacy leakage. Normally, input data **x** is considered extractably memorized if an adversary can construct data that makes the model produce **x** [53]. The relationship between model memorization and privacy leakage can be expressed as:

$$\text{increase}(\mathcal{L}) \sim \text{increase}(\mathcal{M}), \tag{9}$$

where $\mathcal{M}$ is the model memorization and $\mathcal{L}$ denotes the privacy leakage. An increased capacity for the model to learn from input data correlates with a higher likelihood of privacy leakage. With Equations (8) and (9), we have:

$$\mathcal{L}^T > \mathcal{L}^C, \tag{10}$$

where $\mathcal{L}^T$ is the privacy leakage in Transformers, and $\mathcal{L}^C$ is the privacy leakage in CNNs. In summary, Transformers, employing attention modules and featuring a larger receptive field, inherently enhance model memorization, thereby contributing to increased privacy leakage.

B.2 The Impact of the Design of Activation Layers

In Sections 5.2, 5.3 and 6.2, we show that removing activation layers could make the model more vulnerable to privacy attacks. Here we delve into more theoretical discussions.

ReLU vs GELU. ReLU and GELU are two widely adopted activation layers in CNNs and Transformers, defined as follows:

$$\mathrm{ReLU}(\mathbf{x}) = \max(0, \mathbf{x}), \tag{11}$$

$$\mathrm{GELU}(\mathbf{x}) = 0.5\mathbf{x}(1 + \tanh(\sqrt{\frac{2}{\pi}}(\mathbf{x} + 0.044715\mathbf{x}^3))), \tag{12}$$

where $\mathbf{x}$ is the input of the activation layer.

Both ReLU and GELU introduce non-linearity into the model, enabling the learning of complex and hierarchical representations from input data. However, non-linearity can also bring challenges during model training, such as the vanishing gradient and exploding gradient problems, making it difficult to update model weights and biases [39]. The ReLU unit may suppress half of its inputs by outputting zero values, selectively passing information to the next layer [50]. Moreover, the non-differentiability of ReLU at $\mathbf{x} = 0$ poses issues in gradient-based optimization. The GELU unit, designed as a smooth approximation to ReLU, preserves the non-linearity of the model [39]. The adoption of GELU can enhance the model's robustness and generalization, making the adversary extract less information from data samples and decrease the attack performance [4]. This aligns with our experimental results presented in Step 9 of Table 8 and Figure 8.

The removal of activation layers. As using GELU does not help with improving the privacy attack performance, our analysis shows that removing activation layers and leaving fewer activation layers is the key to increasing the model's vulnerability to privacy attacks (shown in Figure 8). This is because retaining fewer activation layers allows the model to preserve more information learned from the training data, resulting in increased privacy leakage compared to models with a greater number of activation layers, which can be written as $\mathcal{L}^{\text{fewer act}} > \mathcal{L}^{\text{more act}}$.

B.3 The Impact of the Design of Stem Layers

In Sections 5.2, 5.3 and 6.2, we illustrate that the design of stem layers (the "Patchify" step) can result in increased privacy leakage. Here, we provide more theoretical discussions.

The stem layers in a standard ResNet have a 7 × 7 convolution layer with stride 2. The "Patchify" strategy introduced by Transformers uses a 4 × 4 convolution layer with stride 4, which is actually a non-overlapping convolution process to make patches of the input data. Some research [17, 23] theoretically show that non-overlapping patches can easily learn information from input data, comparable to the capabilities of overlapping patches. Empirical studies [78] provide further support for the efficacy of this kind of stem layer design with convolutional layers. This design is shown to enhance optimization stability and improve the adversary's attack performance, which is demonstrated in our experimental results in Sections 5.2 and 5.3. We can formulate this as $\mathcal{L}^{\text{patchify stem}} > \mathcal{L}^{\text{resnet stem}}$.

B.4 The Impact of the Design of LN Layers

In Sections 5.2, 5.3 and 6.2, we show that changing BatchNorm to LayerNorm can contribute to increased privacy leakage. Here, we provide more theoretical discussions on the implications of this design choice.

Let $\mathbf{x} = (x_1, x_2, ..., x_H)$ and $\mathbf{y}$ be the input of size H and output of a normalization layer. LayerNorm re-centers and re-scales the input x as

$$\mathbf{y} = \mathbf{g} \odot \frac{\mathbf{x} - \mu}{\sigma} + \mathbf{b}, \mu = \frac{1}{H}\sum_{i=1}^{H} x_i, \sigma = \sqrt{\frac{1}{H}\sum_{i=1}^{H}(x_i - \mu)^2}, \tag{13}$$

where $\odot$ denotes a dot production operation, μ and σ are the mean and standard deviation of $\mathbf{x}$, bias $\mathbf{b}$ and gain $\mathbf{g}$ are learnable parameters [3, 80].

The bias and gain are designed for affine transformation on normalized vectors to enhance performance. However, these parameters are trained based on the training data and may not adequately consider the input distributions of testing data. This increases the risk of overfitting when applying LayerNorm in the model [80]. The overfitting of the model would lead to more privacy leakage of the training data. As a result, models with LayerNorm could potentially expose sensitive information during privacy attacks, which can be written as $\mathcal{L}^{\text{ln layer}} > \mathcal{L}^{\text{bn layer}}$.